Reimagining Postman's Onboarding

By - Frankie Ye

AGENDA

01 Goal

Who do we focus on?
Prioritization
Defining success

02 Onboarding today

Review of existing flow
Baseline measurement
User actions and analytics

03 Experimentation

Identifying business
Strategy and Approach
Designing the experiment

04 Long term vision

Execution, mindset and strategy

05 Appendix

Qualitative research

01

Goal

Thought process

- What are some areas of onboarding that could be optimized for today that would have most business impact tomorrow?
- Which segment of users should we focus on?
    - Free vs Paid
        - If paid: Basic vs Professional vs Enterprise
    - Technical roles vs Non-technical roles
    - First time Postman users vs returning Postman users
- To determine which segments of users to focus on, I'd ideally need the data on user journey during onboarding broken down into different roles and payment plans (which is not available to me).
- Still require a form of prioritization

Postman's pricing plans

Free

$0

Get Started

For individuals or a small team of 3 or less to start testing APIs.

What's included:

- Up to 3 collaborators
- Access to API Client
- Cloud-based integrations
- 3 Packages
- 1 day collection recovery
- 25 collection runs
- Access to Flows
- 50 free activities in Postbot

Available add-ons:

- Collection Runner
- Postbot

Basic

$14 per user/month, billed annually

Buy Now

Basic API collaboration for a single team.

Everything in Free, plus:

- Unlimited collaborators
- 30 day collection recovery
- 10,000 Mock Server requests
- 10,000 Monitor requests
- Additional pay-as-you-go capabilities

Available add-ons:

- Postbot
- Collection Runner
- Flows

Professional

$29 per user/month, billed annually

Buy Now

API collaboration for larger teams, cross-org, and external partners.

Everything in Basic, plus:

- Private Workspaces
- Single & Multi-Partner Workspaces
- Basic role-based access control
- 25 Packages
- 90 days collection recovery

Available add-ons:

- Postbot
- Collection Runner
- Partner Editor
- Flows

Enterprise

$49 per user/month, billed annually

Buy Now | Contact Sales

Org-wide API development with advanced support, security, and control.

Everything in Professional, plus:

- Private API Network
- SSO, SCIM, & SAML
- Reporting & Analytics
- Audit Logs
- User Groups
- Advanced role-based access control
- 100 Packages
- 100,000 Mock Server requests
- 10,000 Monitor requests
- Customer Success

Available add-ons:

- Postbot
- Collection Runner
- Partner Editor
- API Builder
- Advanced Security Administration

Thoughts and assumptions

- I assume that Free tier would have the highest volume of users with lowest price/user and the opposite for the highest paid tier: the enterprise users.
- Assumption: User onboarding flows are the same across free and paid tier members.
- I also want to acknowledge that there needs to be alignment within the organization on which segments we want to grow.

Estimated revenue breakdown

Tier	Estimated % of users	Number of users (of 35M)	Cost per user/mth	Revenue generated annually)
Free	50%	17.5M	$0	-
Basic	30%	10.5M	$14	$1.764B
Professional	20%	7M	$29	$2.436B
Enterprise	10%	3.5M	$49	$2.058B

Source: Postman reported 35M users in 2024

Decision Process

- Free tier users have the highest number of users with lowest
- In 2024, Postman reported 35M users. Source
- Assuming a 5% increase in users from one tier to another here's the additional revenue generated per year, we see that free tier upgrades would bring in the most potential via our onboarding efforts.

Tier	Number of users (of 35M)	5% user upgrade	Additional revenue per user/mth	Additional revenue generated annually)
Free	17.5M	875,000	$14	$147M
Basic	10.5M	525,000	$15	$95M
Professional	7M	350,000	$20	$84M

Goal and success metric

- The ultimate goal is to **convert more users from Free to paid version**. (% of users who converted after 30 days). However Free to Paid user conversion is a **lagging** metric.

- Therefore, I'd use **time to value (TTV) as a success metric** measurement. Today's onboarding process walks users through creating their first API request using Postman's example endpoint. Therefore, I'll **assume that the TTV is relatively similar across all segments and personas.**

- Challenging the assumption:

    - **Perceived value can differ from one user persona to another** and therefore I will assign different user actions for different personas.

    - For example, according to some of the qualitative data. (Refer to **appendix A** for full details.)

    - Intermediate to Advanced level users chose Postman because:

        - The ability to collaborate with other members
        - Ability to create and import collections
        - Of the security features such as not exposing env variables

    - Common Issues (Friction):
        - Had trouble with authentication during first time use

Defining Success - by persona

Skill level	Roles	Value action	TTV Metric
Little API knowledge (Existing)	Technical Writers, UIUX developers, CEO, CIO, Educators etc	Creating API requests	Median time from sign up to first **own-API** request made
Knowledgeable	Software Architects, developers, Dir engineering, Testers	Create/import collection Create workspace Authentication set up	Median time from sign up to 1. Create collection 2. Set up workspace 3. Invite team member 4. Authentication set up (security)

Note: There are much more complex and personalized flows we can build here. For example, roles will also determine the buying decision power which affects the PQL quality.

Company size is also another factor to consider, the user persona could be CEO but it could be a very small team of 2-3 people whereas backend dev could come from a large org which affects the PQL quality.

02

Onboarding today

Fullstack Dev Journey - Free/ New User

Step 1:

Prospect searches for "Postman free trial" on Google search

Step 2:

Prospect lands on /downloads

Step 3:

Prospect signs up for a free account web version.

Step 4:

In-app onboarding: User finished filling up the form on "Tell us about yourself page"

Step 5:

In-app onboarding: User selects "skip for now" on Join your existing teams page

Step 6:

In-app onboarding: User selects "Get started" on the popup modal for "Welcome, let's get started"

Step 7:

In-app onboarding: User is presented with a documentation panel on the right on how to send an example request. User is also provided with example endpoint from Postman for testing.

User then copy, paste and sent a request in the URL bar.

Step 8:

In-app onboarding: User sees the JSON body fetched from the endpoint. The popup modal also updated to step 2 which shows users how to save the request.

User selects "show me how" via the popup modal CTA

Step 9:

In-app onboarding: A popup modal pointing to the floppy disc with save shows up.

User press "save" via the popup modal. And proceeded to save it to a new collection.

Step 10:

In-app onboarding: Left panel opens up with the new collection in which the request was saved. Getting started popup modal changed to step 3 which presents information on Postbot.

User the selects "show me how"

Step 11:

In-app onboarding: Postbot popped up on the right hand side and explained it's capabilities. The let's get started popup modal shows message "Good job you crushed it" indicating that onboarding is now complete, and presenting more on variables, templates , automation and documentation

User the selects "Add variables "

Step 12A: Add variables

In app onboarding: Presented with documentation on how to add variables on the right hand panel

Step 12B: Documentation

In-app onboarding: Presented with documentation on how to document APIs

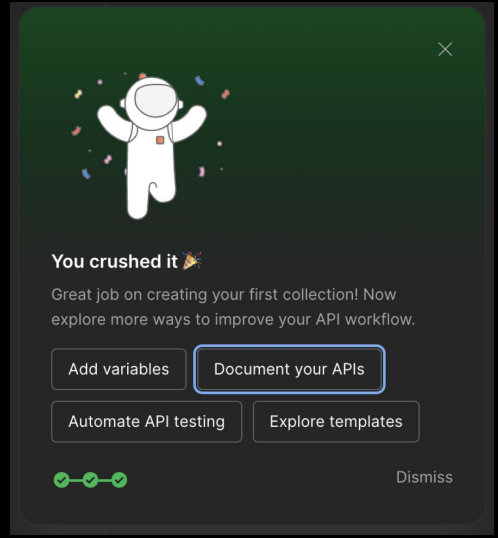
Step 12C: Existing templates

In-app onboarding: Shows a popup modal of templates users can leverage, also categorized based on role, recommendation, use cases

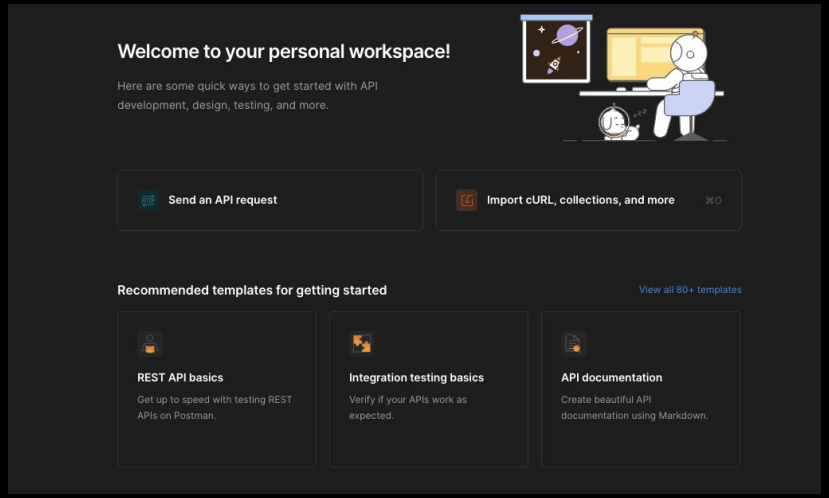
Step 12C: Automate API testing

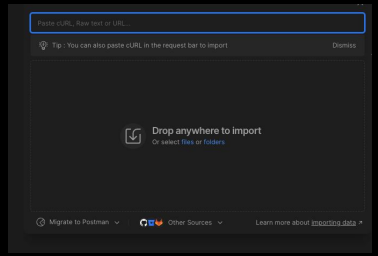
In-app onboarding: Postman documentation panel popsup

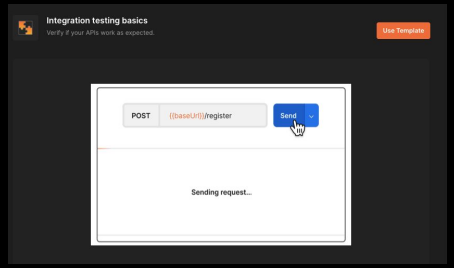
User clicks "dismiss" on the onboarding popup

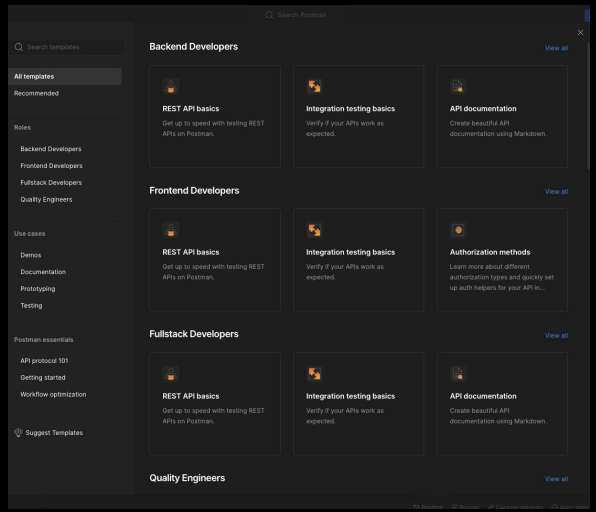

You crushed it 🎉
Great job on creating your first collection! Now explore more ways to improve your API workflow.
Add variables
Document your APIs
Automate API testing
Explore templates
Dismiss

Backend Dev journey - Free/ Returning


Welcome to your personal workspace!
Here are some quick ways to get started with API development, design, testing, and more.
Send an API request
Import cURL, collections, and more
⌘O
Recommended templates for getting started
View all 80+ templates
REST API basics
Get up to speed with testing REST APIs on Postman.
Integration testing basics
Verify if your APIs work as expected.
API documentation
Create beautiful API documentation using Markdown.


Paste cURL, Raw text or URL...
Tip: You can also paste cURL in the request bar to import
Dismiss
Drop anywhere to import
Or select files or folders
Migrate to Postman
Other Sources
Learn more about importing data





Tracking and Analytics

I'd set up tracking to measure where the opportunity areas are and to use it as baseline to compare against and to understand how many users successfully completed onboarding.

Marketing Efforts

- Downloads Landing page visitors
- Form completion rate

In-app onboarding efforts

1. Tell us about yourself page completion
2. Joining team page completion/skips
3. Button click "Get started"
4. Button click "Send" on URL bar
- Button click "Show me how" in popup - Save to collection step
- Button click "save" after
- Button click "Show me how" - Postbot step
- Button click "Add variables" - Congratulations step
- Button click "Documentation"
- Button click "Existing templates"
- Button click "Automate testing"

Tracking and analytics

Additional metrics to track

- Median time from sign up to first own-API request made

- Median time from sign up to (whichever the user first complete)
  - Create/import their own collection
  - Set up workspace
  - Invite team member
  - Authentication set up (security)

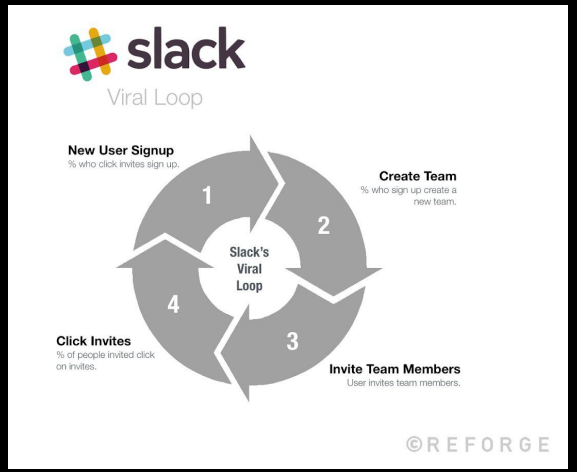
03

Experimentation

Strategy and approach

- The strategy we're using here is to approach onboarding as **persona based onboarding.** Figma is an example of a successful company that deployed this strategy.

- Additionally, in the case of hypothesis 2, we're creating a **viral growth loop**, by introducing user to workspace features during onboarding. This also dovetails into users eventually upgrading from free to paid users as their team sizes grow which contributes to organic growth within the app. Slack and box are such examples that deployed the loops in their product.



Hypothesis 1:

By introducing an onboarding flow for users with limited knowledge that includes steps for them to use their own API endpoint to make a request within the app,

it will decrease the median time from sign up to creating their own API request by x%,

because it will help them get over friction points such as authentication and jumpstart their API related work.

Hypothesis 2:

By introducing an onboarding flow for new users with sufficient knowledge that includes steps for them to use their own API collection/ create workspace/ invite team members

it will decrease their ttv by y%,

because it will help them discover features that are perceived as more valuable for them.

Execution

Parameters to consider when crafting out the experiments

- How should we test this?
- A/B? Time -based analysis (before vs after)?
- Do we care about speed or accuracy?
- We risk running 50% of the traffic to losing variant. So do we have capability run multi-armed bandit (MAB) test?
- Do we have the infrastructure to conduct feature based A/B test (feature flagging)?

Execution

Here's my suggestion and assumptions:

- How should we test this?
    - We should carry these experiments out as A/B test for utmost accurate success measurements. That depends on availability of these 2 factors
        - Experimentation Infrastructure: We have the supporting infrastructure to run a true AB just for this feature.
        - Time frame: True AB would mean traffic would be split 50-50 between test vs control group which also means that it'll take some time for the experiment to reach statistical significance. However it'll eliminate seasonality affecting our test results and come to a more reliable conclusion.

04

Long term vision & scaling

Long term and scaling

- Leverage additional parameters such as company size, buying decision power (from roles) in the onboarding flow to prioritize leads.
    - Since we're only allowing users to sign up using their work emails we can determine the company sizes of these users.
- Onboarding once vs ever-onboarding approach: Adopt the approach of **ever-onboarding to improve retention as well as expansion** as product adds more features and users get more advanced.
- Localization: Localize product onboarding by geo-location of the user

05

Appendix

Appendix A - User Research

Video 1: Postman Review: Postman Review: An incredible tool for professionals and amateurs alike

- Date: 2 years ago
  Skill level: Novice
- Role: VP of CS
- Was using: Chrome extension
- Need: Something robust
- Onboarding and integrate into your business: A little hard for someone who doesn't have prior development experience - Opportunity for novice level onboarding

Appendix A - User Research

Video 2: Postman Review: https://www.youtube.com/watch?v=GH4chE7rjfo

- Date: 2 years ago
- Skill level: Intermediate - Advanced
- Role: I.T Consultant
- Reasons for choosing Postman:
    - Likes being able to share workspaces with colleagues
    - Likes being able to import collections from another copy of Postman

- How easy was the onboarding and integrating into your business
    - Most of the fields are intuitively named
    - Got tricky when it comes to authentication (Had to rely on documentation)

- What recommendations do you have for others considering?
    - Do you need to test authorization and authentication
    - Do you need to collaborate

Appendix A - User Research

Video 3: Postman Review:
https://www.youtube.com/watch?v=W7kgOpp8u3Y

- Date: 4 years ago
- Skill level: Advanced
- Role: Software Architect
- Was using: ARC (Advanced res client)
  - lack of creating collections ,
  - One request at time
  - No way to set env variables
  - need to be able to share with teams
- Why choose collections
  - Easily create collections out of request
  - Import call commands
  - Import collection through a URL
  - Create a call command out of postman request
  - Postman allows you to create env variable (without exposing passwords)
- How easy was onboarding and integrating?
  - 3 or 4 clicks for installation
  - Does not require admin access to your laptop
- What recommendations do you have for others considering?
  - He recommended it's preferable for REST based and not so for SOAP
  -